www.bt-magazin.de 2.2010 Deutschland: € 14,90 Österreich: € 16,80 Schweiz: sFr 29,80 Luxemburg: € 16,90

Business Technology

Business Technology

Architektur & Management Magazin

Expertenwissen für IT-Architekten, Projektleiter und Berater

Stark:
„Der Kritiker sorgt für die Qualität“

AGILITÄT

Business Case für Agilität

Retrospektiven – wider die Macht der Verdrängung

Flexible Architek-turen

4 191858 414906 02

Lösungsmuster, Teil 2

Enterprise SOA Security

AUTOREN: DR. DIRK KRAFZIG, JOST BECKER, OLIVER MAHNKE UND ILJA PAVKOVIC

KEIN GRUND ZUR VERZWEIFLUNG

Nachdem wir im ersten Teil unserer Serie die Notwendigkeit eines geänderten Securityverständnisses aufgezeigt haben, stellen wir im vorliegenden zweiten Teil existierende und praktikable Lösungsmuster vor. Dazu werden wir uns mit dem zugrunde liegenden Basisprinzip der Security-Tokens und darauf aufsetzenden Mechanismen wie Single Sign-on (SSO), frühe und späte Berechtigungsprüfungen sowie verschiedenen Aspekten von Domain-übergreifender Security befassen.

DIE GRUNDPRINZIPIEN TRADITIONELLER SECURITY

In existierenden großen IT-Landschaften dominieren bis heute Anwendungssilos, also Applikationen, deren vertikale Struktur gegebenenfalls mehrere Schichten besitzt (Multi-Tier), die in ihrer horizontalen fachlichen Ausdehnung aber weitgehend isoliert sind. Da Security gewöhnlich den Zugriff auf Daten regelt und fast alle Applikationen ihre Daten vorwiegend in Datenbanksystemen verwalten, wird in der Silowelt Security oftmals über Datenbanksessions gelöst. Eine Datenbanksession stellt eine temporäre Verknüpfung zwischen Anwendung und Datenbank bzw. Ressource her. Die Session beginnt typischerweise mit dem expliziten Login des Benutzers in die Datenbank und endet mit dem Logout. In dem Zeitraum vom Login bis zum Logout besteht eine direkte Verbindung zwischen Ressource und der nutzenden Applikation.

Vielfach sind Middleware-Komponenten wie der Treiber des jeweiligen Relationalen Datenbanksysteme in den Anwendungen verbaut, d. h. entsprechende Module werden im Build-Prozess oder zur Laufzeit eingebunden. Nicht selten *kennt* das Datenbanksystem den Endnutzer, weil Identität, Rollen und Rechte des Anwenders parallel zu den fachlichen Daten anwendungsspezifisch abgelegt sind.

Der Start einer Session wird auf Basis eines Protokolls ausgehandelt, das spezifisch für die zu nutzende Datenbank ist. Anschließend *lebt* die Session eingebettet in der proprietären Middleware. Sessions zwischen verschie-

Vierteilige Reihe: Enterprise SOA Security

Teil 1: Herausforderungen	Der erste Teil befasst sich mit den Herausforderungen heutiger IT-Landschaften. Abteilungsübergreifende, integrierte Lösungen und die Öffnung von Kernsystemen für Kunden und Lieferanten überfordern häufig traditionelle Sicherheitslösungen
Teil 2: Lösungsmuster	Im zweiten Teil der Serie werden Lösungsmuster vorgestellt, mit deren Hilfe moderne SOA Security konzipiert werden kann. Im Zentrum der Diskussion stehen Security-Tokens, die in einer verteilten Umgebung sicherstellen, dass jede Komponente gesicherte Annahmen über ihre Nutzer und deren Berechtigungsprofile treffen kann
Teil 3: Web-Services-Standards	Zahlreiche Web-Services-Standards wie SAML, WS Security, XACML oder WS Trust helfen bei der Umsetzung interoperabler Sicherheitslösungen in einer SOA. Der dritte Teil gibt einen Überblick über existierende Standards und wie sie in der Praxis angewandt werden
Teil 4: Organisatorische Maßnahmen	SOA Security ist keinesfalls ein reines Technologiethema. Abteilungsübergreifende Prozess- und Rollenkonzepte erfordern auch abteilungsübergreifende Governance und ein Umdenken in den Risikoabteilungen großer Unternehmen

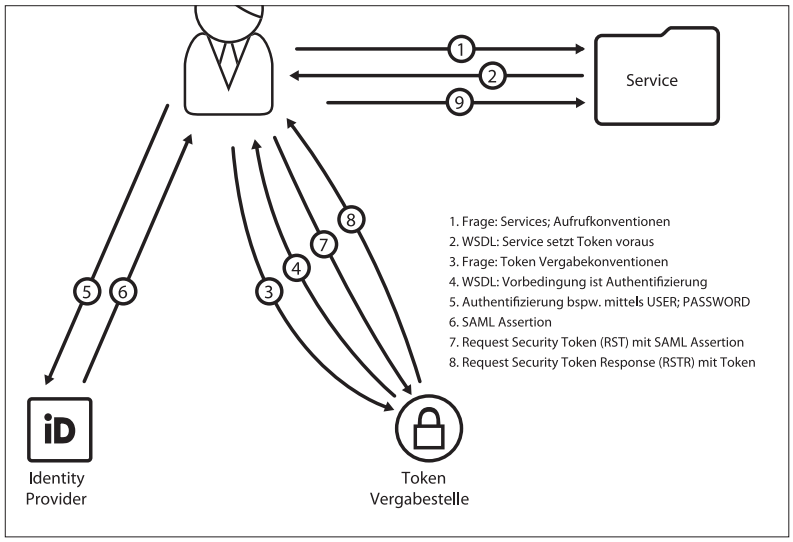

Abb. 1: Securityprolog – neun Schritte bis zum fachlichen Service

denen Plattformen sind in der Regel nicht kompatibel. Dieses Prinzip funktioniert, solange man sich innerhalb der definierten Grenzen der eigenen Anwendung bewegt. In einem Integrationsszenario – also im Rahmen der zunehmenden Bestrebungen, einzelne Anwendungen in einen größeren, Abteilungs- und sogar Unternehmensgrenzen überwindenden Kontext einzubinden und dazu die seitlichen, *fachlichen* Grenzen zu überwinden – bereitet diese Vorgehensweise aber große Schwierigkeiten. Weder kennt die Anwendung die Sicherheitsmechanismen der anderen, *fremden* Ressourcen, noch ist sie mit der richtigen Middleware ausgestattet, die eine Verbindung überhaupt erst ermöglicht. Außerdem sind die Anwendungsnutzer bei den *fremden* Ressourcen unbekannt und damit ohne Berechtigung.

Wie bereits erwähnt, sind diese Probleme natürlich nicht unüberwindlich. In der Praxis existieren zu derartigen Problemfällen häufig Workarounds. Wenn aber Integration nicht die Ausnahme ist, sondern zur Regel wird, dann führt eine größer werdende Menge von Patches und Workarounds zunehmend zu einer Anwendungslandschaft, die aufgrund ihrer Komplexität weder beherrschbar ist noch Sicherheit als übergreifendes, zusammenhängendes Konzept umsetzt.

DIE GRUNDPRINZIPIEN VERTEILTER SECURITY

Um Security in verteilten Systemen zu gewährleisten, wird das Sessionkonzept durch das Konzept des Security-Tokens ersetzt. Grundgedanke dabei ist die Existenz einer von Anwendung und Ressource unabhängigen Instanz – der Token-Vergabestelle. Diese stellt der Ressource ein so genanntes Security-Token zur Verfügung. Das Security-Token bündelt Zusicherungen über eine Menge von Eigenschaften des Anfordernden, die als Claims bezeichnet werden. Aus diesen Zusicherungen lassen sich in der Folge Berechtigungen ableiten. Dieses Konzept ist insbesondere in einer serviceorientierten Architektur unverzichtbar, in der verteilte Service-Provider und Service-Consumer an die Stelle traditioneller Anwendungen bzw. Ressourcen treten. Die Etablierung des Security-Token-Konzepts unterstützt das SOA-Prinzip der losen Kopplung maßgeblich, da auf anwendungsspezifische Sessions und eine damit einhergehende enge technische Bindung der beteiligten Kommunikationspartner verzichtet werden kann.

Damit das Security-Token seine Aufgabe erfüllen kann, muss es definierten Standards genügen, da nur dadurch die Möglichkeit einer formellen Überprüfung besteht. Außerdem muss das Token von allen Ressourcen akzeptiert werden, was die Bekanntheit und die Vertrauenswürdigkeit der erwähnten Instanz – der Token-Vergabestelle oder auch des Security Token Service (STS) – bedingt.

Wenn eine Anwendung als Service-Consumer Teile ihrer Funktionalität mittels Aufrufen von Services realisiert, dann geht den eigentlichen fachlichen Serviceaufrufen ein Sicherheitsprolog voraus, der den Austausch von Security-Tokens beinhaltet. Dabei sind häufig mehrere Akteure beteiligt – z. B. der Herausgeber der Identity, die Token-Vergabestelle oder gegebenenfalls der Herausgeber von Zertifikaten. Als kurzen Überblick listet Tabelle 1 die wichtigsten Bausteine zur Umsetzung verteilter Security auf. Diese für Enterprise SOA Security notwendigen Bausteine werden wir im Folgenden durch die konkrete Beschreibung des Basisprozesses sowie die Auflistung der wichtigsten darauf aufsetzenden Lösungsmuster exemplarisch ausweiten.

Im ersten Teil unserer Artikelserie hatten wir bereits eine wichtige Grundidee vorgestellt, nämlich dass die Überprüfung einzelner Identitätsmerkmale ausreichend ist, um sicherheitsrelevante Freigaben zu erteilen. Am Beispiel des Service *Weinhandel* haben wir demonstriert, wie der Service-Consumer (Kunde) einen Service des Service-Providers (Händler) aufruft, das Ergebnis aber erst übermittelt wird, nachdem der Provider ein von einer dritten, vertrauenswürdigen Instanz (Ausweisvergabestelle) bestätigtes Identitätsmerkmal (Geburtsdatum bzw. Mindestalter für den Kauf alkoholhaltiger Getränke) überprüft hat. Wichtig ist, dass im Sinne eines schlanken Prozesses und unter Einhaltung der Prinzipien der Datensparsamkeit

und Datenvermeidung – einem durchaus wichtigen Aspekt bei öffentlichen und größeren Kunden, siehe §3a Bundesdatenschutzgesetz – möglichst wenig Identitätsinformationen preiszugeben, nur die relevanten Identitätsmerkmale des Konsumenten überprüft werden und deshalb auch gar nicht alle zur Überprüfung bereitgestellt werden müssen.

Im Rahmen einer SOA-Security-Implementierung nimmt ein STS die Rolle der vertrauenswürdigen Instanz ein. Dem Aufruf eines Service geht eine Folge von Kommunikationsschritten zwischen Service-Consumer, Service-Provider, Identity-Provider, Token-Vergabestelle und gegebenenfalls dem Herausgeber von Zertifikaten voraus. Wir werden nicht näher auf die kryptografische Absicherung der einzelnen Verbindungen eingehen, sondern uns auf einen gängigen Prozessablauf konzentrieren. Es sei der Vollständigkeit halber darauf hingewiesen, dass SOA Security auch auf Basis anderer Standards und/oder Formate umgesetzt werden kann. Der Consumer bezieht über eine Service-Registry eine mittels Web Services Description Language (WSDL) formulierte Beschreibung des Service, den er nutzen möchte. Sie beinhaltet als Teil des Servicevertrags Informationen über die angebotenen Operationen und – idealerweise als separate Anlage – Anforderungen, die der Provider an seine Consumer stellt, die so genannten Policies. Sowohl sicherheitsrelevante Zusicherungen, die der Provider von Consumern verlangt, als auch das konkrete Format, in dem diese übertragen werden müssen, sind in Form von Policies in der Servicebeschreibung enthalten. So kann der Consumer vor einer Nutzung des Service z. B. nachweisen, eine bestimmte Altersgrenze überschritten zu haben.

Ein gängiges Format zur Formulierung von Zusicherungen ist die Security Assertion Markup Language (SAML). Diese definiert, wie auf der Grundlage von XML Zusicherungen als so genannte SAML-Token beschrieben und übertragen werden. Einen Wert hat eine derartige Zusicherung für den Provider aber nur dann, wenn diese von unabhängiger, dritter Stelle bekräftigt wurde. Der Consumer wendet sich an einen vom Service-Provider akzeptierten Security Token Service (STS), dessen Adresse er z. B. aus einer Registry beziehen kann. Analog zur Kommunikation mit dem Provider werden Aufrufkonventionen verhandelt, die u. a. vorgeben, wie sich der Consumer beim STS zu authentifizieren hat. Beispielsweise kann der STS einen Identitätsnachweis von einem ihm vertrauensvoll bekannten Identity-Provider fordern, der entweder öffentlich, in der Organisation des Consumers oder aber von einem dritten Betreiber zur Verfügung ge-

Token	Ein Token ist ein Textfragment, das Informationen bzw. Zusicherungen über den Consumer enthält. Tokens werden in den Header fachlicher Nachrichten eingebettet. Der formale Aufbau und die Position des Tokens innerhalb der Nachrichten sind in entsprechenden Standards definiert. Die im Token enthaltenen Identitätsmerkmale werden individuell festgelegt. Dazu gehören auch Vereinbarungen, ob die Token-Information verschlüsselt und/oder signiert zu übertragen ist.
Identity Provider	Der Identity Provider ist ein Herausgeber von Identitäten. Er kann extern sein oder zur Organisationseinheit des Service-Consumers gehören. Ihm obliegt das Identitätsmanagement, d. h. er verwaltet Nutzer und deren Daten, bestätigt auf Anforderung die Existenz eines Nutzers und sichert diesem bestimmte Eigenschaften zu.
Token-Vergabestelle	Bei der Token-Vergabestelle handelt es sich um eine unabhängige, vertrauenswürdige Instanz, die in der Lage ist, auf Anfrage ein Token mit Zusicherungen über einen Nutzer zu erstellen. Üblicherweise steht die Token-Vergabestelle als Service unternehmensintern oder webweit unter einer bekannten Adresse (URI) zur Verfügung. Die Token-Vergabestelle und der Identity Provider arbeiten eng zusammen. Sie können Module eines einzigen Systems sein oder durch Austausch von Zertifikaten in einer Vertrauensbeziehung stehen. Synonyme für Token-Vergabestelle sind Token Service oder auch Security Token Serivce (STS).
Regel-Engine zur Berechtigungsprüfung	Der Service-Provider entscheidet auf Basis der übergebenen Zusicherungen, ob einem Consumer die Servicenutzung erlaubt ist – oder eben auch nicht. Die dazu nötigen Regeln können fest durch den Provider implementiert sein bzw. optional von einer so genannten Regel-Engine ausgewertet werden. Darunter versteht man ein Softwaremodul, das auf geeignetem Abstraktionsniveau aus übergebenen Zusicherungen und einem Regelwerk Berechtigungen erteilt oder verweigert.
Herausgeber von Zertifikaten	Um zwischen Komponenten einer verteilten Architektur Vertrauensbeziehungen zu etablieren und eine sichere Datenübertragung zu gewährleisten, werden üblicherweise asymmetrische kryptografische Verfahren, so genannte Public-Key-Verfahren, eingesetzt. Mittels digitaler Signaturen kann die Korrektheit des Absenders einer Nachricht und deren Integrität überprüft werden. Die Gültigkeit der dabei verwendeten Schlüssel wird durch Zertifikate nachgewiesen. Das setzt eine Infrastruktur für die Erzeugung, Verteilung und Verwaltung von Zertifikaten (Public-Key-Infrastruktur, PKI) voraus. Zertifikate können unternehmensintern – zur Absicherung unternehmensweiter Kommunikation – oder aber extern von öffentlichen Institutionen vergeben werden.
Service-Registry	Eine Service-Registry ist ein zentrales Verzeichnis von Service-Providern. Sie liefert Consumern auf Anfrage Informationen über diese, z. B. die konkrete Adresse eines Service-Providers oder die Anforderungen, die dieser an einen Consumer stellt.
Service-Provider	Ein Service-Provider ermöglicht Consumern den Zugriff auf eine Ressource über eine standardisierte Schnittstelle.

Tabelle 1: Bausteine verteilter Security

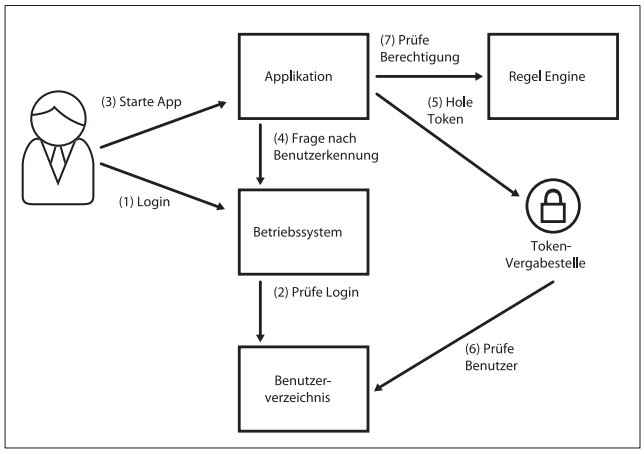

Abb. 2: Ablaufschema Single Sign-on (SSO)

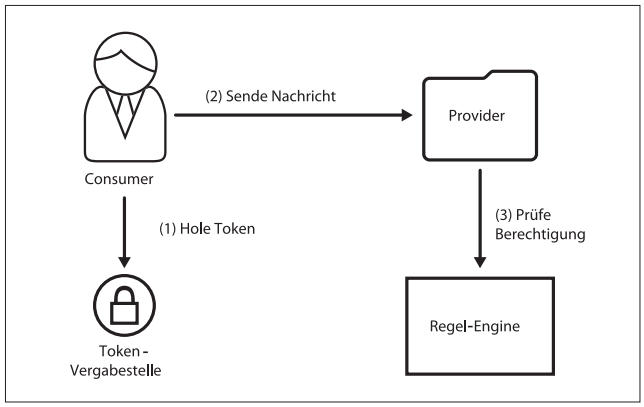

Abb. 3: Späte Berechtigungsprüfung durch den Provider

stellt wird – z. B. OpenID von Google oder Yahoo (**Abb. 1**). Kann der Consumer alle geforderten Informationen und Formate beibringen, erhält er vom STS ein signiertes Token, das er wie gewünscht in die Kommunikation mit dem Service-Provider einbinden kann.

LÖSUNGSMUSTER 1: SSO

Single Sign-on (SSO) erlaubt es Anwendern, sich zu Beginn einer Sitzung bzw. eines Arbeitstages einmalig anzumelden und anschließend mit den verschiedensten Anwendungen – auch solchen, die normalerweise ein eigenes Login erfordern – zu arbeiten, ohne sich immer wieder authentifizieren zu müssen. Single Sign-on ist nicht nur eine Erleichterung für Anwender, sondern bietet ebenso viele Vorteile für Administratoren und Sicherheitsverantwortliche. Jede Anwendung, die ein zusätzliches Login und damit ein zusätzliches Passwort verlangt, verursacht Aufwand, der durch SSO vermieden werden kann. Zum einen reduziert sich der Aufwand für den Anwender durch die Vermeidung mehrfach durchzuführender Anmeldungsprozeduren. Zum anderen können die zuständigen Administratoren Zugriffsberechtigungen global und konsistent aktivieren oder auch deaktivieren, sollte sich z. B. die Rolle eines Mitarbeiters ändern oder dieser aus dem Unternehmen ausscheiden. Typische Sicherheitslöcher wie *vergessene Logins* ehemaliger Anwender können so vermieden werden. Gerade SSO bedarf in der Umsetzung einer modernen verteilten Security, damit die zentrale Verwaltung der Identitäten nicht zu einem zentralen Sicherheitsloch gerät (**Abb. 2**).

Durch SSO lassen sich darüber hinaus noch weitergehende Szenarien umsetzen. Beispielsweise ist es möglich, dezentral verwaltete Identitäten separater Domänen unter einer zentral verwalteten Identität, einem so genannten Master-Account, zusammenzufassen. Die Domänen können z. B. einzelnen Fachbereichen eines Unternehmens entsprechen, unterschiedlichen Trägern einer föderalen Organisationsstruktur oder auch eigenständigen Unternehmen, die Geschäftsbeziehungen über Unternehmensgrenzen hinweg serviceorientiert abbilden. Die einzelnen Identitäten unterscheiden sich typischerweise in Eigenschaften wie Kennung, Passwort, Rechnungsadresse etc. und sind nur für die jeweilige lokale Domäne gültig. In diesem Szenario, das auch als Identity Federation bezeichnet wird, liefert ein globaler STS im Zusammenspiel mit lokalen Identitätsprovidern oder STS der einzelnen Domänen einen globalen Identitätsnachweis.

LÖSUNGSMUSTER 2: SPÄTE BERECHTIGUNGSPRÜFUNG

Die vom Security Token Service gegebene Zusicherung ist nur ein Teil der Berechtigungsprüfung. Den weitaus wichtigeren Teil machen die Regeln aus, anhand derer entschieden wird, ob die Zusicherungen für eine Berechtigung auf einen Service ausreichen. Ein Grundmuster für SOA Security ist, diese Regelüberprüfung möglichst spät – d. h. vom Service-Provider – durchführen zu lassen. Voraussetzung dafür ist insbesondere ein vorher erfolgreich abgeschlossenes SSO-Verfahren (**Abb. 3**). Eine in der Praxis häufig angewandte Variante der Berechtigungsprüfung sieht gegebenenfalls noch einen weiteren Informationsaustausch zwischen Regel-Engine und Token-Service vor.

Grundsätzlich können Tokens für eine maximale, umfassende Menge von Claims vergeben werden oder aber als Mini-Token für eine deutlich kleinere Menge von Basis-Claims. Mit dem umfangreichen Token stehen dem Service-Provider alle Zusicherungen zur Verfügung, die ihn interessieren könnten – allerdings gewiss auch einige, die ihn eigentlich nicht interessieren. Dafür ist dann aber keine weitere Kommunikation zwischen den Parteien notwendig, d. h. es gibt keine

weitere Verzögerung. In der zweiten Variante, der Zusicherung einer möglichst kleinen Anzahl von Claims, kann es dazu kommen, dass der Provider weitere Zusicherungen benötigt. Das bedingt zusätzliche Kommunikation und zieht damit den Sicherheitsprolog in die Länge. Dafür werden nur die Zusagen transferiert, die minimal für die Autorisierung der Servicenutzung notwendig sind. Nachteil neben der zusätzlichen Kommunikation zwischen der Regel-Engine und dem Token-Service ist zusätzliche Komplexität. Dafür werden nur die wirklich notwendigen Daten ausgetauscht, d. h. dem nicht auf SOA eingeschränkten Securitygrundsatz „So viel Informationen wie nötig, so wenig wie möglich" wird besser Genüge geleistet.

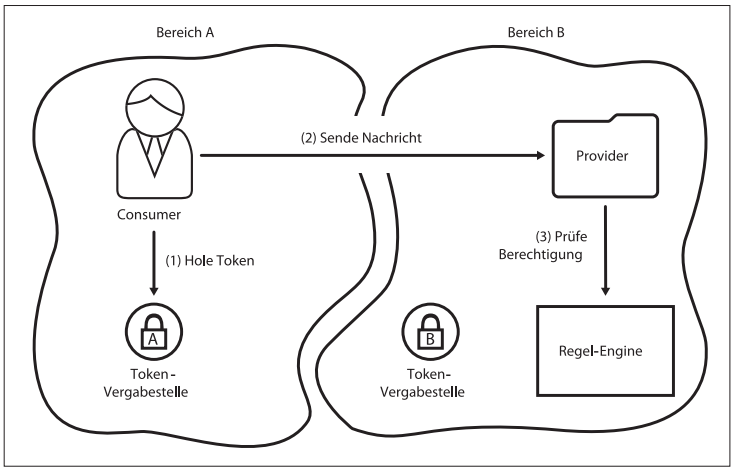

Abb. 4: Beispiel zweier Security-Domains mit zwei Token-Vergabestellen

LÖSUNGSMUSTER 3: FRÜHE BERECHTIGUNGSPRÜFUNG

Bei der frühen Berechtigungsprüfung wird hingegen eine komplett andere Philosophie verfolgt, nämlich die Bereitschaft, den Consumer in den Berechtigungsvorgang einzubinden. In diesem Ansatz besorgt sich der Consumer zunächst ein Token mit seinen eigenen Eigenschaften. Mit diesem Token fragt der Consumer bei der Regel-Engine an, ob er einen von ihm gewünschten Service aufrufen darf. Die Regel-Engine kreiert ein neues Token, das die Autorisierungsentscheidung enthält, signiert seine Antwort und sendet sie an den Consumer zurück. Der Consumer ruft anschließend den Provider auf. Sofern der Provider der Signatur der Regel-Engine vertraut, führt er den Aufruf gemäß der von der Regel-Engine getroffenen Autorisierungsentscheidung aus. Eine Variante dieses Ansatzes ist, dass nicht die Regel-Engine ihre Antwort in ein Token generiert, sondern den Token-Service verwendet, um sein Antwort-Token zu verpacken. Dieser Ansatz hat den Vorteil, dass die Regel-Engine nur einmal aufgerufen werden muss, wenn anschließend viele gleichartige Aufrufe durchgeführt werden sollen, z. B. wenn ein Consumer beabsichtigt, sukzessive mehrere Adressen eines Kunden einzutragen. Der Nachteil dieser Vorgehensweise liegt in der Tatsache, dass der Provider nur von speziellen, *intelligenten* Konsumenten aufgerufen werden kann, die auf die speziellen Kommunikationsanforderungen reagieren können. Dadurch ist diese Technik üblicherweise auf den Einsatz innerhalb von Unternehmensgrenzen beschränkt.

LÖSUNGSMUSTER 4: CROSS-DOMAIN SECURITY MIT KOMPATIBLEM TOKEN

Selbst innerhalb der Grenzen eines Unternehmens ist keineswegs sichergestellt, dass es nur eine Securityinfrastruktur gibt. Häufig gibt es in der Praxis tatsächlich mehrere Unternehmensbereiche, die aufgrund von Firmenakquisitionen, Mergern, gewünschten Redundanzen oder schlicht ihrer schieren Größe über mehr als einen STS – und damit über mehr als eine Security-Domain – verfügen (**Abb. 4**).

Im einfachsten Fall kennt der Bereich B die Sprache, in der die Claims des Bereichs A formuliert sind. Aus fachlicher Sicht bedeutet das, dass beispielsweise die Namen der vergebenen Rollen bekannt sind. Technisch setzt das die Kenntnis der gemeinsam bzw. gegenseitig genutzten Standards und Verfahren voraus, unter anderem, wie Tokens aufgebaut sind – beispielsweise in SAML 2.0. Damit der sichere Kommunikationsfluss in einer derartigen Konstellation gewährleistet bleibt, ist der Austausch von Zertifikaten notwendig, d. h. Bereich B muss über ein Zertifikat aus Bereich A verfügen, damit entsprechend zertifizierten Nachrichten aus dem Bereich A auch im Bereich B das notwendige Vertrauen entgegengebracht wird.

LÖSUNGSMUSTER 5: CROSS-DOMAIN SECURITY MIT TOKEN-TRANSFORMATION BEIM CONSUMER

Nicht immer sind auszutauschende Tokens kompatibel. Vielfach existiert in der Praxis die Notwendigkeit, Tokens zwischen verschiedenen Standards oder unterschiedlichen Versionen eines Standards zu transformieren. Dabei müssen – im obigen Verständnis – sowohl fachliche als auch technische Transformationen durchgeführt werden. Diese Aufgabe fällt einem oder mehreren so genannten Gateways zu. Ein Gateway muss natürlich die eingesetzten Standards, die fachlichen Zusammenhänge bzw. Abbildungsvorschriften und die Zertifikate beider Welten kennen. Eine Consumer-nahe Token-Transformation kommt beispielsweise dann in Betracht, wenn es in einem B2B-Netzwerk viele verschiedene Consumer gibt und der Provider *seinen* Standard durchsetzen kann.

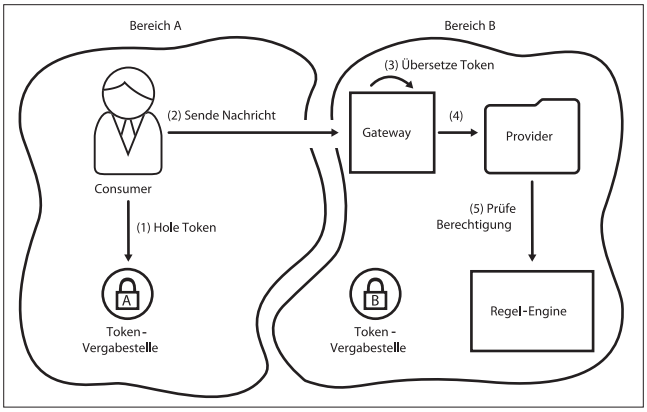

Abb. 5: Providernahes Gateway ist verantwortlich für Token-Transformation

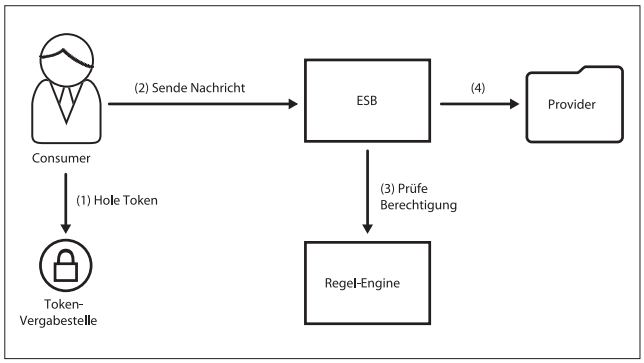

Abb. 6: ESB ist verantwortlich für Berechtigungsprüfung

LÖSUNGSMUSTER 6: CROSS-DOMAIN SECURITY MIT TOKEN-TRANSFORMATION BEIM PROVIDER

Die elegantere Lösung für die Transformation von Tokens zwischen Domains stellt die providernahe Lokalisierung des Gateways dar. Damit wird ein *Securityeingangstor* für eine abgeschlossene Domain etabliert, über das alle externen Anfragen in Bereich B kanalisiert werden (**Abb. 5**). Diese Variante bietet weitere Vorteile: In Kombination mit Firewall-Mechanismen kann die Sicherheit erhöht werden. Die Bündelung der Zugriffe im Gateway erlaubt ein zentralisiertes, aussagekräftiges Logging und Auditing und eine sinnvolle Ausnahmenbehandlung, und trägt so der Tatsache Rechnung, dass mehrere Geschäftsbereiche in die Kommunikationsbeziehungen involviert sind. Man kann eine Umsetzung von externer verschlüsselter Kommunikation zu unverschlüsselter interner Kommunikation vornehmen. Das kann insbesondere dann interessant sein, wenn eine hohe CPU-Bandbreite für das Kodieren/Dekodieren der Nachrichten benötigt wird, die an zentraler Stelle mit speziell ausgerüsteten Kryptoprozessoren bereitgestellt werden kann. Dort kann auch ein Abrechnungsmodul den externen Aufrufern die Kosten für den Service in Rechnung stellen u. v. m.

LÖSUNGSMUSTER 7: ESB-BASIERTE SECURITY

In der ESB-basierten Security wird der Provider von der Berechtigungsprüfung befreit. In diesem Szenario muss jeder securityrelevante Aufruf durch den ESB geleitet werden. Der ESB bestimmt die Adresse des Providers und nimmt vor der Weiterleitung des Aufrufs die Berechtigungsprüfung vor. Nur Anfragen, für die der Consumer autorisiert ist, werden zum Provider durchgelassen. Damit der Provider sichergehen kann, dass er nur autorisierte Anfragen erhält, verwendet der ESB ein spezielles Token, das seine Identität zusichert. Erhält der Provider eine Anfrage mit diesem Token, führt er diese Anfrage ohne weitere Prüfung durch (**Abb. 6**).

ESB-basierte Security hat sehr prägnante Vor- aber auch Nachteile: Als Vorteil ist zu werten, dass die Service-Provider schlanker werden, da der Aspekt der Berechtigungsprüfung nicht von jedem Service-Provider einzeln, sondern zentral über den ESB umgesetzt wird. Auf diese Weise ist eine zentrale und standardisierte Zugriffskontrolle möglich. Weiterhin ist ESB-basierte Security leichter zu administrieren als eine verteilte Berechtigungsprüfung, da sowohl die Policies der Service-Provider als auch die zur technischen Realisierung der Berechtigungsprüfung nötigen Middleware-Komponenten an zentraler Stelle konfiguriert und überwacht werden können. Das Konzept ist einfach und als solches den an der Umsetzung von Security Beteiligten leicht zu kommunizieren. Dem stehen jedoch gewichtige Nachteile gegenüber: Durch die Einführung eines ESB wird eine starke zentrale Abhängigkeit geschaffen. Häufig sind ESB-Produkte als zentrale Middleware-Komponente realisiert, über die jeglicher Kommunikationsfluss erfolgen muss. In vielen Fällen sind mit einer solchen Lösung Anforderungen an die Dienstgüte eines Service-Providers nicht oder nur mit hohem Aufwand umsetzbar. Falls Service-Provider bzw. -Consumer und der zentrale ESB beispielsweise regional stark verteilt sind, führt die Kommunikation über den ESB zu erhöhten Latenzzeiten, sodass ein bestimmtes Antwortzeitverhalten oder ein gewünschter kritischer Durchsatz möglicherweise nicht erreicht werden kann.

Um zu vermeiden, dass durch den ESB ein zentraler Flaschenhals entsteht, ist häufig die Anschaffung teurer Serverkomponenten nötig. Auch die Implementierung der ESB-Security muss hohen Qualitätsanforderungen entsprechen, damit diese im Betrieb nicht zu Performanceeinbußen führt. Kommerzielle ESB-Produkte verursachen zudem häufig hohe Lizenzkosten, deren Rechtfertigung zumeist nicht einfach durch einen Business-Case abgesichert werden kann. Die zentralisierte Umsetzung von Security wertet überdies die Stellung der Service-Provider ab. Diese sind weniger autark, sodass das zentrale

SOA-Prinzip der losen Kopplung nur in geringerem Maße realisiert werden kann.

ZUSAMMENFASSUNG UND AUSBLICK

Sicherheit kann auch in verteilten Systemen auf Enterprise-Niveau aufrechterhalten werden. Zentrales Konzept ist die Vergabe von Security-Token, die einem Service-Provider als verlässliche Zusicherungen securityrelevanter Eigenschaften der Consumer dienen. Darauf basierende Lösungsmuster wie Single Sign-on, Berechtigungsprüfungen mittels Regel-Engine, Token-Transformation oder Übertragung der Securityverantwortung an einen ESB ermöglichen angemessene Reaktionen auf unterschiedlichste Gefährdungssituationen.

Im kommenden dritten Artikel (BT 3.10) werden wir den Autorisierungsprozess mithilfe von Tokens auf definierten Standards wie WS Trust, WS Policy, WS SecurityPolicy etc. beleuchten und die wichtigsten Standards beschreiben. Im abschließenden vierten Artikel (BT 4.10) widmen wir uns dann ausführlich den organisatorischen Konsequenzen, die die zielorientierte Einführung belastbarer Security innerhalb einer SOA mit sich bringt. Denn SOA ist mehr als das Zusammenfügen geeigneter Techniken und Werkzeuge. Wesentliche Bestandteile für die erfolgreiche Einführung und den späteren Betrieb einer SOA sind die unter dem Oberbegriff „Governance“ zusammengefassten zu etablierenden Kommunikations- und Verantwortungsstrukturen. Deren Notwendigkeit gilt natürlich auch für den Teilaspekt Security innerhalb einer SOA. Zur effektiven Umsetzung werden von der Unternehmensleitung autorisierte Ansprechpartner benötigt, die den Projekten mit Vorgaben und Umsetzungsunterstützung zur Seite stehen und den querschnittlichen Betrieb der Securitysystemkomponenten verantworten.

Dr. Dirk Krafzig ist Gründer von SOAPARK. Als Sprecher auf Konferenzen und Autor von Artikeln und Büchern gilt Dr. Krafzig als ein Protagonist der serviceorientierten Architektur (SOA) und hat maßgeblich zu der Begriffsbildung in diesem Bereich beigetragen. Insbesondere die SOA-Fallstudien mit der Deutschen Post, Credit Suisse, Halifax Bank of Scotland und Winterthur Versicherung in seinem Bestseller „Enterprise SOA, Prentice Hall, 2004.“ haben viel Aufmerksamkeit auf sich gezogen. Derzeit arbeitet Dr. Krafzig in einem strategischen SOA-Programm bei einem Mobilfunkanbieter an dem Thema Security.

Jost Becker ist gemeinsam mit Ilja Pavkovic und Oliver Mahnke Gründer der binaere bauten gmbh. Sein Interesse gilt parallel zu informationstechnologischen Themen seit jeher ökonomischen und organisatorischen Aspekten im Unternehmen – eine Kombination, die sich in seinem aktuellen Tätigkeitsfeld als Berater für Software- und IT-Unternehmensarchitekturen auszahlt. Aktuell richtet er seinen Fokus auf die speziellen Securityanforderungen in verteilten Systemen.

Oliver Mahnke ist Chefarchitekt der binaere bauten gmbh. In dieser Rolle verantwortet er sowohl Anwendungs- als auch Unternehmensarchitekturen. Derzeit erarbeitet er die zentralen Architekturrichtlinien für eines der größten deutschen Versicherungsunternehmen.

Ilja Pavkovic hat langjährige praktische Erfahrung in strategischen und global aufgestellten IT- Projekten. In wechselnden Rollen als Projektleiter und Softwarearchitekt bildeten generative Softwareentwicklung und Security immer wieder Schwerpunkte. Insbesondere kennt er auch traditionelle Securitylösungen und ihre Grenzen.

Links & Literatur

[1] Bundesamt für Sicherheit in der Informationstechnik: SOA-Security-Kompendium, Version 2.0, 2009